#include<stdio.h>

#include<math.h>

#include<conio.h>

#include<stdlib.h>

#include<string.h>

#define N 10000 *//Decalring the maximum limit for input.*

int add1(int num1[], int num2[], int ans[], int n1, int n2){

*//here n1 and n2 are the lengths of respective numbers*.

int i, j, k, temp, carry, add; //add variable carries out the addition of one particular index.

*/\**

*Here the basic logic is that as we have numbers stored in an array in reverse manner,*

*we can add one by one and store it in an differnt array (that too in a reverse way)*

*and carry the 'carry' to the next addition.*

*Here we return the length of the answer array.*

*\*/*

carry=0; *//keeping carry variable to carry to the next addition*.

k=0;

i=0;

while(i<n1 || i<n2){

add=num1[i]+num2[i]+carry;

if(add>=10){

ans[k++]=add-10; *//to check if the carry is generated. If it does then we subtract 10 and carry one to carry variable.*

carry=1;

}

else{

ans[k++]=add;

carry=0;

}

i++;

}

if(carry==1) *//To check if the last digits of two numbers have a carry. Then we can add it to the answer array.*

ans[k++]=carry;

return k; *//returning the length of the array.*

}

int sub(int num1[], int num2[], int ans[], int n1, int n2){

int i, j, k, temp, carry, diff;

*/\**

*Here the basic logic is to carry out the subtraction one by one adn if the carry the subtraction is negative then*

*we take a carry from the next index and then we will have to assume one extra of the previous indexed number.*

*\*/*

carry=0;

k=0;

i=0;

while(i<n1 || i<n2){ *//we run the loop till both of them get exhausted!*

if(num1[i]>=num2[i]){

diff=num1[i]-num2[i]-carry;

ans[k++]=diff;

carry=0;

}

else{

diff=10+num1[i]-num2[i]-carry; *//to check if the carry is required.*

ans[k++]=diff;

carry=1;

}

i++;

}

return k; *//returning the length of the answer array.*

}

int mul(int num1[], int num2[], int ans[], int n1, int n2){

int i, j, k, temp1, carry, diff;

*/\**

*Here we do the multiplication like we do in pen and paper method.*

*we multiply each number with the other complete number and put a zero at the next index and*

*add all those numbers totally.*

*example:*

*4 6 9 8*

*\* 2 3 3*

*-----------------*

*1 4 0 9 4*

*+ 1 4 0 9 4 0*

*+ 9 3 9 6 0 0 <---------This is the extra zero we add. & hence we skip it while adding it again.*

*-------------------------*

*1 0 9 4 6 3 4*

*Due to those extra zeroes, while addiing to the answer array again what we do is to skip that particularly and*

*carry out the addition through the next index.*

*\*/*

j=0;

k=0;

int y=-1; *//Number of indices to skip!*

*/\*Here the carry variable denotes the carry of multiplication while the carry1 denotes the carry of addition.\*/*

while(j<n2){

carry=0;

int carry1=0;

y++;

k=y;

for(i=0; i<n1; i++){

int sum=(num2[j]\*num1[i]+carry)%10;

carry=(num2[j]\*num1[i]+carry)/10;

if(ans[k]+sum>=10){

ans[k]=ans[k]+sum-10+carry1;

carry1=1;

}

else{

ans[k]=ans[k]+sum+carry1;

carry1=0;

}

k++;

}

if(ans[k]+carry>=10){

ans[k]=ans[k]+carry-10+carry1;

carry1=1;

k++;

}

else{

ans[k]=ans[k]+carry+carry1;

carry1=0;

k++;

}

if(carry1==1){

ans[k]=carry1;

k++;

}

j++;

}

return k; *//returning the length of the answer array*.

}

int division(int num1[], int num2[], int n1, int n2){

int ans[2\*N]={0}, mod[N]={0}, i, j, k, temp, carry, add, div, flag1;

*/\**

*Here the logic is very simple.*

*We continously add the divisor till we reach the divident.*

*Here we make use of multiplication function.*

*And to find modulo the same function can be used.*

*We subtract the divident from answer and store it back to the answer.*

*And again subtract divisor from the answer to get modulo!*

*\*/*

int n=n2; *//initialisation for n*.

int flag=0;

div=0;

while(!flag){

div++;

n=add1(num2, ans, ans, n, n);

if(n>n1){

div*--; //here we decrease the quoetiont by one to make the answer correct else we would have answer wrong. (CHECK!)*

flag=1;

}

else if(n1==n){

flag1=0;

for(i=n-1; i>=0 && !flag1; i--){

if(ans[i]>num1[i]){

flag=1;

}

else if(ans[i]<num1[i]){

flag1=1;

}

}

if(flag)

div--;

}

}

*/\*To check the modulus!\*/*

*n=sub(ans, num1, ans, n, n1);*

*n=sub(num2, ans, ans, n2, n);*

*printf("\nThe modulo is ");*

for(i=n-1; i>=0; i--){

printf("%d", ans[i]);

}

printf("\n");

return div;

}

int main(){

char numStr1[N], numStr2[N];

int num1[N]={0}, num2[N]={0}, ans[2\*N]={0}, i, k, temp, n, choice, flag=0;

while(!flag){

*/\*We take the inputs as string inputs.*

*And convert them to array of integers using ASCII values.*

*We Store the integers in reverse fashion in the array so that*

*the further calculations will be easier.*

*\*/*

printf("Enter the first BIG INTEGER & second BIG INTEGER:\n");

scanf("%s", numStr1);

scanf("%s", numStr2);

int n1=strlen(numStr1);

int n2=strlen(numStr2);

*/\*We store the digits in revrse order so as to help during operations.\*/*

k=0;

for(i=n1-1; i>=0; i--){

temp=numStr1[i]-48;

num1[k++]=temp;

}

k=0;

for(i=n2-1; i>=0; i--){

temp=numStr2[i]-48;

num2[k++]=temp;

}

printf("\nEnter the operation you want to perform.\

\n1.Addition of two BIG INTEGERS.\

\n2.Subtraction of two BIG INTEGERS.\

\n3.Multiplication of two BIG INTEGERS.\

\n4.Division of two BIG INTEGRS.\

\n5.To stop the operations.\n");

scanf("%d", &choice);

switch(choice){

case 1:

n=add1(num1, num2, ans, n1, n2);

printf("The required answer is ");

for(i=n-1; i>=0; i--){

printf("%d", ans[i]);

}

printf("\n");

break;

case 2:

flag=0;

if(n2>n1){

flag=1;

}

else if(n1==n2){

int flag1=0;

for(i=n1-1; i>=0 && !flag1; i--){

if(num2[i]>num1[i]){

flag=1;

}

else if(num2[i]<num1[i]){

flag1=1;

}

}

}

if(!flag)

n=sub(num1, num2, ans, n1, n2);

else

n=sub(num2, num1, ans, n2, n1);

int i=n-1;

printf("The required answer is ");

if(flag)

printf("-");

while(ans[i]==0){

i--;

}

for(; i>=0; i--){

printf("%d", ans[i]);

}

printf("\n");

break;

case 3:

n=mul(num1, num2, ans, n1, n2);

printf("The required answer is ");

for(i=n-1; i>=0; i--){

printf("%d", ans[i]);

}

printf("\n");

break;

case 4:

n=division(num1, num2, n1, n2);

printf("The required answer is %d", n);

printf("\n");

break;

case 5:

flag=1;

break;

default:

printf("Wrong entry. Please select again.");

break;

}

}

return 0;

}